UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/676,227	09/30/2003	Gunter Schwarzbauer	06896/100M097-US1	9707

7278 7590 07/14/2006

DARBY & DARBY P.C.
P. O. BOX 5257
NEW YORK, NY 10150-5257

EXAMINER	
RUTLEDGE, AMELIA L	

ART UNIT	PAPER NUMBER
2176	

DATE MAILED: 07/14/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/676,227	SCHWARZBAUER ET AL.	
	Examiner	Art Unit	
	Amelia Rutledge	2176	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 April 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-31* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-31* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to communications: Amendment, filed 04/13/2006.

2. Claims 1-31 are pending in the case. Claims 1 and 28-31 are independent claims.

Information Disclosure Statement

Regarding the Information Disclosure Statement filed 04/08/2004, the references cited in the International Search Report have been considered, and they were separately listed in the Information Disclosure Statement. However, the International Search Report itself, was lined out and not marked as considered on the IDS, because the International Search Report itself is not prior art. The references cited in the International Search Report and IDS (AA, AB, and AC) were considered by the examiner on 01/05/2006 and will be made of record and identified on the face of any patent which may issue from this application.

Claim Rejections - 35 USC § 101

3. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

4. Claims 1-28 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Regarding independent claim 1, claim 1 is directed to computer related nonstatutory subject matter, since although it claims functional descriptive material in

the form of a system for automatic context management, it is not recorded on a computer readable medium. Rather, claim 1 is directed to software *per se*, and is therefore not statutory. See *Warmerdam*, 33 F.3d at 1361, 31 USPQ2d at 1760.

In regard to dependent claims 2-27, claims 2-27 are rejected because they add nothing to render the claimed subject matter statutory.

In regard to independent claim 28, claim 28 is directed to computer related nonstatutory subject matter, because it claims nonfunctional descriptive material, instead claiming abstract ideas, for example: comparing form data and generating data based on differences resulting from the comparing step, etc. Thus, no requisite functionality is present to satisfy the practical application requirement. See *Diehr*, 450 U.S. at 185-86, 209 USPQ at 8. In addition to being directed to nonfunctional descriptive material, the method of claim 10 is not recorded on a computer readable medium, and for this reason is also not statutory since it is not capable of causing functional change in the computer.

Claim Rejections - 35 USC § 112

5. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

6. Claims 9-12 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter

which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

7. Claims 9-11 have been amended to add the phrase *each of said parser additions of said library of parser additions*. For example, claim 9 cites: *The system according to claim 9, wherein each of said parser additions of said library of parser additions implements an algorithm for parsing hyperlinks from an HTML document*. The specification does not disclose a library of parser additions wherein *each* parser addition in the library of parser additions implements the algorithms as claimed in claims 9-11.

8. Regarding dependent claim 12, claim 12 is dependent on claim 9 and is rejected for incorporating the deficiencies of its base claim.

Claim Rejections - 35 USC § 102

9. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

10. Claims 1-12 and 29 are rejected under 35 U.S.C. 102(e) as being anticipated by Weinberg et al. (hereinafter "Weinberg"), U.S. Patent No. 6,549,944 B1, issued April 2003.

Regarding independent claim1, Weinberg teaches an automated testing tool for testing and monitoring web applications (Col. 2, l. 30 – Col. 3, l. 37; Col. 8, l. 40-67); compare to *A system for automatic context management for testing, monitoring and automating a network application having client side executable code.* Weinberg teaches that the testing tool has a Dynamic Scan feature for scanning dynamic HTML code, i.e., client side executable code, which may be set to automatically fill in dynamic forms without interaction from a user (Col. 23, l. 15-Col. 26, l. 19, especially col. 24, l. 1-33); compare to *a tool operable to parse said client side executable code so as to determine a subsequent state of the network application free of interaction with a user.* Weinberg teaches that the testing tool has a recorder operable to record a context full load testing script of a test scenario, and a replay engine to execute the test script (Col. 32, l. 23- Col. 33, l. 37). Weinberg teaches that the recorder and replay engine have a context full API, since the test scenario is generated using information specific to the site context, i.e., server access log files and associated site usage information (Col. 33, l. 40-Col. 35, l. 20).

Regarding dependent claim 2, Weinberg teaches a page-level API (Col. 19, l. 11-50; Fig. 8).

Regarding dependent claims 3-6, Weinberg teaches an extensible document parser operable to determine at least one parser extension, where the parser extension includes a replay instruction specifying at least one parser addition and parameters for the parser addition, since Weinberg teaches a variety of plug-in modules for the document parser (Col. 17, l. 53-60), including the Action Tracker and Load Wizard

containing replay instructions specifying parser additions; as well as parser extensions in the form of other external client applications (Col. 18, l. 40-59).

Regarding dependent claims 7-10, Weinberg teaches a library of parser additions, since Weinberg teaches the addition multiple plug-ins, each of which implements a specific parsing algorithm (Col. 17, l. 53-Col. 18, l. 20), including an algorithm for parsing hyperlinks, the Link Doctor. Weinberg teaches an algorithm for parsing HTML forms, the Dynamic Scan feature (Col. 25, l. 50-Col. 26, l. 51), including different configurations, i.e., extensions, of the Dynamic Scan feature including a standard web browser or a special integrated web browser (Col. 26, l. 41-51).

Regarding dependent claim 11, Weinberg teaches a parser addition which implements an algorithm for parsing embedded documents from an HTML document, since Weinberg teaches a Dynamic Scan feature for scanning dynamic HTML code, i.e., client side executable code, which may be set to automatically fill in dynamic forms without interaction from a user (Col. 23, l. 15-Col. 26, l. 19, especially col. 24, l. 1-33); and the forms were embedded in HTML documents.

Regarding dependent claim 12, Weinberg teaches a method of parsing hyperlinks by searching text between a left and right boundary string (Col. 20, l. 62-Col. 21, l. 60); it is inherent in the teaching of Weinberg that this method would have been used by the plug-ins, since Weinberg teaches an API with objects including information about the URLs and links (Col. 19, l. 11-25).

Regarding independent claim 29, Weinberg teaches an automated testing tool for testing and monitoring web applications (Col. 2, l. 30 – Col. 3, l. 37; Col. 8, l. 40-67);

including a processor and memory (Col. 7, l. 51-Col. 8, l. 12; claim 13). Weinberg teaches that the testing tool has a Dynamic Scan feature for scanning dynamic HTML code, i.e., client side executable code, which may be set to automatically fill in dynamic forms without interaction from a user (Col. 23, l. 15-Col. 26, l. 19, especially col. 24, l. 1-33); compare to *said tool operable to parse said client side executable code so as to determine a subsequent state of the network application free of interaction with a user.* Weinberg teaches that the testing tool has a recorder operable to record a context full load testing script of a test scenario, and a replay engine to execute the test script (Col. 32, l. 23- Col. 33, l. 37). Weinberg teaches that the recorder and replay engine have a context full API, since the test scenario is generated using information specific to the site context, i.e., server access log files and associated site usage information (Col. 33, l. 40-Col. 35, l. 20).

Claim Rejections - 35 USC § 103

11. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

12. Claims 13-28, 30, and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Weinberg, as applied to claims 1-12 and 29 above, in view of Sidles, U.S. Pub. No. 2002/0062342, published May 2002.

Regarding dependent claims 13 and 14, Weinberg teaches that the API comprises form replay instructions, since Weinberg teaches that the testing tool has a Dynamic Scan feature for scanning dynamic HTML code, i.e., client side executable code, which may be set to automatically fill in dynamic forms without interaction from a user, when replayed (Col. 23, l. 15-Col. 26, l. 19, especially col. 24, l. 1-33). Weinberg does not explicitly teach form merging instructions including instructions for merging an HTML form and one context-full script form to produce a form to be submitted. However, Sidles teaches a reference to a previously downloaded form in a test script, i.e., to see if the user made changes to a field, and instructions for merging the form and scripts containing previously submitted form information to automatically fill out a form to be submitted (p. 12, par 120-123). Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding dependent claim 15, while Weinberg does not explicitly teach instructions for merging each individual form field, Sidles teaches merging instructions for each individual form field of the HTML form and script form (p. 10-11, par. 102-104). Sidles teaches instructions for matching each field of the form to a dictionary entry for the form, and if a match cannot be found, the field information is not sent, compare to

an instruction to send a form field value obtained from said HTML form; an instruction to send a form field value specified in said script form; and an instruction to not send one of said form fields. Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding dependent claim 16, Weinberg teaches associating a database query with a URL of a site and automatically performing the query and mapping the results next time an automatic update is performed (Col. 24, l. 1-19); compare to *an action URL in said test script instead of an action URL obtained from said HTML form for said form to be submitted.*

Regarding dependent claim 17, Weinberg recording a page level test script comprising parser extensions, since Weinberg teaches that the testing tool has a recorder operable to record a context full load testing script of a test scenario, and a replay engine to execute the test script (Col. 32, l. 23- Col. 33, l. 37), and since Weinberg teaches a variety of plug-in modules for the document parser (Col. 17, l. 53-60). Weinberg does not explicitly teach form merging instructions including instructions for merging an HTML form and one context-full script form to produce a form to be submitted. However, Sidles teaches a reference to a previously downloaded form in a

test script, i.e., to see if the user made changes to a field, and instructions for merging the form and scripts containing previously submitted form information to automatically fill out a form to be submitted (p. 12, par 120-123). Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding dependent claim 18, Weinberg teaches that testing tool has a recorder to record session history by building representations of all web pages downloaded so far during a recording session, because the Dynamic Scan feature records session history and builds representations of all web pages downloaded (Col. 23, l. 15-Col. 26, l. 19). Weinberg also teaches a map representation of recorded pages from a site session.

Regarding dependent claims 19 and 20, Weinberg teaches a page-level API (Col. 19, l. 11-50; Fig. 8). Weinberg teaches an extensible document parser operable to determine at least one parser extension, where the parser extension includes a replay instruction specifying at least one parser addition and parameters for the parser addition, since Weinberg teaches a variety of plug-in modules for the document parser (Col. 17, l. 53-60), including the Action Tracker and Load Wizard containing replay instructions specifying parser additions; as well as parser extensions in the form of other

external client applications (Col. 18, l. 40-59). Weinberg teaches that the recorder utilizes the document parsers. While Weinberg does not explicitly teach that the recorder automatically detects which form merging instructions are needed in order to record a test script which will correctly use dynamic information during a script replay, Sidles teaches an automatic method of form filling where the system tests which rules and instructions are needed in order to correctly fill a dynamic form (p. 12, par 125). Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding dependent claim 21, while Weinberg does not explicitly teach that the recorder detects the need for recording at least one of said parser extensions by detecting that a context-less replay instruction is to be recorded otherwise, Sidles teaches a process performed by the completed form analysis engine for detecting the need for recording parser extensions, i.e., dictionary or database entries (p. 11, par. 107-113) and detecting whether instructions should be recorded with or without context (p. 11, par. 110-112). Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling

system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding dependent claim 22, claim 22 is directed toward substantially similar subject matter as claimed in dependent claims 3, 4, 7, and 8 and is rejected along the same rationale, except where Claim 22 recites the additional limitation*: said recorder is operable to detect which one of said parser extensions is to be recorded by querying each of said parser additions for suitable parameters.* While Weinberg does not explicitly teach the claimed limitation, Sidles teaches recording new rules for forms by querying a history database for suitable parameters (p. 7, par. 88). Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding dependent claims 23 and 24, while Weinberg does not explicitly teach fuzzy form detection, Sidles teaches a method of fuzzy form detection comparing a form being submitted to all forms in a history database (p. 9, par. 86-90), choosing a form from said session history which is most similar to said form being submitted.

Sidles teaches generating logic rules for the form to be used to complete the form fields (p. 9, par. 90), compare to *recording said form merging instructions so that said recorded form merging instructions applied to said form chosen from said session history result in a form identical to said form being submitted.* That is, the dictionary database is adjusted so that the automatic filler program and the rules engine will fill out that form the next time it is encountered (p. 10, par. 91, l. 1-8). Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding dependent claim 25, while Weinberg does not explicitly teach that the replay engine executes a test script with parser extensions and form merging instruction, Sidles teaches an automatic form filling system using fuzzy logic where a history unit generates a new set of rules based on form context (p. 9, par. 88, 90), i.e., *recording at least one context-full test script* for filling the form. Sidles also teaches that a replay engine is capable of executing the test script. Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool

with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding dependent claim 26, claim 26 is directed to substantially similar subject matter as claimed in dependent claim 18, and is rejected along the same rationale.

Regarding dependent claim 27, Weinberg teaches a page-level API (Col. 19, l. 11-50; Fig. 8). Weinberg teaches an extensible document parser operable to determine at least one parser extension, where the parser extension includes a replay instruction specifying at least one parser addition and parameters for the parser addition, since Weinberg teaches a variety of plug-in modules for the document parser (Col. 17, l. 53-60), including the Action Tracker and Load Wizard containing replay instructions specifying parser additions; as well as parser extensions in the form of other external client applications (Col. 18, l. 40-59). Weinberg teaches that the replay engine of the test scenario recorder utilizes the document parsers.

Regarding independent claim 28, Weinberg teaches a method of comparing a form being submitted to one form in a session history, since Weinberg teaches that the testing tool has a Dynamic Scan feature for scanning dynamic HTML code, i.e., client side executable code, which may be set to automatically fill in dynamic forms without interaction from a user (Col. 23, l. 15-Col. 26, l. 19, especially col. 24, l. 1-33) using session history forms. While Weinberg does not explicitly teach generating data based upon differences resulting from the comparing step, for each form in the session history,

or recording form merging instructions for fuzzy form detection, Sidles teaches a method of fuzzy form detection where a form is chosen from a database of forms which is similar to the form being submitted, and merged with past form data to produce a new filled form (p. 6, par. 16-62; p. 9, par. 86-90). Sidles teaches that the comparison and generating steps are performed for each form in the session history, since all forms from the previous sessions are stored in the database for comparison against the submitted form. Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding independent claim 30, Weinberg teaches an automated testing tool for testing and monitoring web applications (Col. 2, l. 30 – Col. 3, l. 37; Col. 8, l. 40-67); including a processor and memory (Col. 7, l. 51-Col. 8, l. 12; claim 13). Weinberg teaches that the testing tool has a Dynamic Scan feature for scanning dynamic HTML code, i.e., client side executable code, which may be set to automatically fill in dynamic forms without interaction from a user (Col. 23, l. 15-Col. 26, l. 19, especially col. 24, l. 1-33); compare to *said tool operable to parse said client side executable code so as to determine a subsequent state of the network application free of interaction with a user.*

Weinberg teaches that the testing tool has a recorder operable to record a context full load testing script of a test scenario, and a replay engine to execute the test script (Col. 32, l. 23- Col. 33, l. 37). Weinberg teaches that the recorder and replay engine have a context full API, since the test scenario is generated using information specific to the site context, i.e., server access log files and associated site usage information (Col. 33, l. 40-Col. 35, l. 20). Weinberg teaches a page-level API (Col. 19, l. 11-50; Fig. 8).

Weinberg teaches an extensible document parser operable to determine at least one parser extension, where the parser extension includes a replay instruction specifying at least one parser addition and parameters for the parser addition, since Weinberg teaches a variety of plug-in modules for the document parser (Col. 17, l. 53-60), including the Action Tracker and Load Wizard containing replay instructions specifying parser additions; as well as parser extensions in the form of other external client applications (Col. 18, l. 40-59). Weinberg teaches a library of parser additions, since Weinberg teaches the addition multiple plug-ins, each of which implements a specific parsing algorithm (Col. 17, l. 53-Col. 18, l. 20), including an algorithm for parsing hyperlinks, the Link Doctor. Weinberg teaches an algorithm for parsing HTML forms, the Dynamic Scan feature (Col. 25, l. 50-Col. 26, l. 51), including different configurations, i.e., extensions, of the Dynamic Scan feature including a standard web browser or a special integrated web browser (Col. 26, l. 41-51). Weinberg teaches a method of parsing hyperlinks by searching text between a left and right boundary string (Col. 20, l. 62-Col. 21, l. 60).

Weinberg teaches recording a page level test script comprising parser extensions, since Weinberg teaches that the testing tool has a recorder operable to record a context full load testing script of a test scenario, and a replay engine to execute the test script (Col. 32, l. 23- Col. 33, l. 37), and since Weinberg teaches a variety of plug-in modules for the document parser (Col. 17, l. 53-60). Weinberg does not explicitly teach form merging instructions including instructions for merging an HTML form and one context-full script form to produce a form to be submitted. However, Sidles teaches a reference to a previously downloaded form in a test script, i.e., to see if the user made changes to a field, and instructions for merging the form and scripts containing previously submitted form information to automatically fill out a form to be submitted (p. 12, par 120-123). Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Regarding independent claim 31, Weinberg teaches an automated testing tool for testing and monitoring web applications (Col. 2, l. 30 – Col. 3, l. 37; Col. 8, l. 40-67); including a processor and memory (Col. 7, l. 51-Col. 8, l. 12; claim 13). Weinberg teaches that the testing tool has a Dynamic Scan feature for scanning dynamic HTML code, i.e., client side executable code, which may be set to automatically fill in dynamic

forms without interaction from a user (Col. 23, l. 15-Col. 26, l. 19, especially col. 24, l. 1-33); compare to *said tool operable to parse said client side executable code so as to determine a subsequent state of the network application free of interaction with a user.* Weinberg teaches that the testing tool has a recorder operable to record a context full load testing script of a test scenario, and a replay engine to execute the test script (Col. 32, l. 23- Col. 33, l. 37). Weinberg teaches that the recorder and replay engine have a context full API, since the test scenario is generated using information specific to the site context, i.e., server access log files and associated site usage information (Col. 33, l. 40-Col. 35, l. 20). Weinberg teaches a page-level API (Col. 19, l. 11-50; Fig. 8).

Weinberg teaches an algorithm for parsing HTML forms, the Dynamic Scan feature (Col. 25, l. 50-Col. 26, l. 51), including different configurations, i.e., extensions, of the Dynamic Scan feature including a standard web browser or a special integrated web browser (Col. 26, l. 41-51). Weinberg teaches a method of parsing hyperlinks by searching text between a left and right boundary string (Col. 20, l. 62-Col. 21, l. 60). Weinberg teaches recording a page level test script, since Weinberg teaches that the testing tool has a recorder operable to record a context full load testing script of a test scenario, and a replay engine to execute the test script (Col. 32, l. 23- Col. 33, l. 37). Weinberg does not explicitly teach form merging instructions including instructions for merging an HTML form and one context-full script form to produce a form to be submitted. However, Sidles teaches a reference to a previously downloaded form in a test script, i.e., to see if the user made changes to a field, and instructions for merging the form and scripts containing previously submitted form information to automatically fill

out a form to be submitted (p. 12, par 120-123). Both Weinberg and Sidles are analogous art, being directed toward the testing and management of web applications. It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Sidles' form filling system with test scripts to Weinberg's automated testing tool with automatic filling of dynamic HTML forms, so that the testing tool of Weinberg would have the benefit of an self learning method of filling out forms which would improve performance on future occasions based on historic results (Sidles, p. 2, par. 17).

Response to Arguments

13. Applicant's arguments with respect to claims 1-31 have been considered but are moot in view of the new ground(s) of rejection. The new grounds of rejection includes the Weinberg patent, which is being relied upon to teach the newly claimed limitations, which include *a tool operable to parse said client side executable code so as to determine a subsequent state of the network application free of interaction with a user* (Claim 1). The Weinberg patent also teaches additional claim limitations which were not disclosed by the Quinlan patent which was relied upon in the previous Office Action, and therefore claims 1-12 and 29 are being rejected under 35 U.S.C. 102(e).

Conclusion

14. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP

§ 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Amelia Rutledge whose telephone number is 571-272-7508. The examiner can normally be reached on Monday - Friday 9:30 - 6:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Heather Herndon can be reached on 571-272-4136. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

AR

Doug Hutton
Primary Examiner
Tech Center 2100